IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicant:	David E. Lowell	§	Art Unit:	2195
		§		
Serial No.:	10/676,922	§		
		§	Examiner:	Eric Charles Wai
Filed:	October 1, 2003	§		
		§		
For:	Runtime Virtualization and Devirtualization of I/O Devices by a Virtual Machine Monitor	§ § §	Atty. Dkt. No.:	200309154-1 (HPC.0518US)

Mail Stop Appeal Brief-Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

SECOND APPEAL BRIEF PURSUANT TO 37 C.F.R § 41.37

Sir:

The final rejection of claims 1-66 is hereby appealed.

I. REAL PARTY IN INTEREST

The real party in interest is Hewlett-Packard Development Company, L.P. The Hewlett-Packard Development Company, LP, a limited partnership established under the laws of the State of Texas and having a principal place of business at 20555 S.H. 249 Houston, TX 77070, U.S.A. (hereinafter "HPDC"). HPDC is a Texas limited partnership and is a wholly-owned affiliate of Hewlett-Packard Company, a Delaware Corporation, headquartered in Palo Alto, CA. The general or managing partner of HPDC is HPQ Holdings, LLC.

II. RELATED APPEALS AND INTERFERENCES

None.

III. STATUS OF THE CLAIMS

Claims 1-66 have been finally rejected and are the subject of this appeal.

IV. STATUS OF AMENDMENTS

No amendment after final rejection has been submitted.

V. SUMMARY OF THE CLAIMED SUBJECT MATTER

The following provides a concise explanation of the subject matter defined in each of the independent claims involved in the appeal, referring to the specification by page and line number and to the drawings by reference characters, as required by 37 C.F.R. § 41.37(c)(1)(v). Each element of the claims is identified by a corresponding reference to the specification and drawings where applicable. Note that the citation to passages in the specification and drawings for each claim element does not imply that the limitations from the specification and drawings should be read into the corresponding claim element.

Independent claim 1 recites in a computer including an I/O device, a method comprising using a virtual machine monitor (Fig. 1:114) to commence virtualization of the I/O device at runtime (Spec., p. 4, ¶ [0020]; p. 5, ¶ [0023]; Fig. 2).

Independent claim 17 recites in a computer including hardware, a virtual machine monitor (Fig. 1:114) running on the hardware (Fig. 1:110), an operating system running on the virtual machine monitor, the hardware including an I/O device, the I/O device already virtualized by the virtual machine monitor, a method comprising devirtualizing the I/O device at runtime (Spec., p. 4, ¶ [0020]; p. 13, ¶ [0045]; p. 14, ¶ [0049]; Fig. 6).

Independent claim 29 recites a computer comprising:

hardware (Fig. 1:110) including an I/O device; and

computer memory (Fig. 1:115) encoded with a virtual machine (Fig. 1:114) for running on the hardware and commencing virtualization of the I/O device at runtime (Spec., p. 4, ¶ [0020]; p. 5, ¶ [0023]; Fig. 2).

Independent claim 38 recites a computer comprising:

hardware (Fig. 1:110) including an I/O device; and

computer memory (Fig. 1:115) encoded with a virtual machine monitor (Fig. 1:114) for devirtualizing the I/O device at runtime (Spec., p. 4, ¶ [0020]; p. 13, ¶ [0045]; p. 14, ¶ [0049]; Fig. 6).

Independent claim 47 recites an article for a computer including an I/O device, the article comprising software for commencing virtualization of the I/O device at runtime (Spec., p. 4, ¶ [0020]; p. 5, ¶ [0023]; Fig. 2).

Independent claim 56 recites an article for a computer including an I/O device, the article comprising computer-readable memory encoded with software for causing the computer to devirtualize the I/O device at runtime (Spec., p. 4, ¶ [0020]; p. 13, ¶ [0045]; p. 14, ¶ [0049]; Fig. 6).

VI. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

A. Claims 1-8, 10, 12-14, 16-19, 25-26, 28-34, 36-38, 44, 46-51, 53-57, 61, 63, and 65-66 Rejected Under 35 U.S.C. § 102(b) as Anticipated by U.S. Patent Publication No. 2004/0117532 (Bennett).

B. Claims 9, 11, 15, 20-24, 27, 35, 39-43, 45, 52, 58-62, and 64 Rejected Under 35 U.S.C. § 103(a) as Unpatentable Over Bennett Alone.

VII. ARGUMENT

The claims do not stand or fall together. Instead, Appellant presents separate arguments for various independent and dependent claims. Each of these arguments is separately argued below and presented with separate headings and sub-headings as required by 37 C.F.R. § 41.37(c)(1)(vii).

A. Claims 1-8, 10, 12-14, 16-19, 25-26, 28-34, 36-38, 44, 46-51, 53-57, 61, 63, and 65-66 Rejected Under 35 U.S.C. § 102(b) as Anticipated by U.S. Patent Publication No. 2004/0117532 (Bennett).

1. Claims 1-8, 10, 12, 13.

It is respectfully submitted that independent claim 1 is not anticipated by Bennett. Independent claim 1 recites a method that uses a virtual machine monitor to **commence** virtualization of an I/O device of a computer **at runtime**.

Although Bennett discloses use of a virtual machine monitor (VMM) that "may emulate and export a bare machine interface to higher level software" (Bennett, ¶ [0015], lines 7-9), Bennett does not provide any teaching that its VMM commences virtualization of an I/O device of a computer at runtime. In fact, it appears that the VMM of Bennett is much like the VMM described in ¶¶ [0004]-[0006] of the background section of the present application, which describe a conventional VMM that virtualizes a hardware from bootup to shutdown, which incurs overhead even when virtualization is not necessary. Specification, ¶ [0006].

The Examiner cited the following passages of Bennett as purportedly disclosing the subject matter of claim 1: ¶¶ [0023], [0053], and [0054]. Paragraph [0023] of Bennett states that the VMM sets the values of the interrupt control indicators before transferring control to a virtual machine. Later, in ¶ [0026] of Bennett, Bennett explains that if an interrupt is generated during the operation of guest software, the appropriate interrupt control indicator is consulted to determine whether the interrupt is to be managed by guest software or by the VMM. These teachings of Bennett do not have anything to do with using a VMM to commence virtualization of the I/O device at runtime.

Paragraph [0053] of Bennett refers to a preferred virtual machine that manages interrupts in the system, and a non-preferred virtual machine that does not manage interrupts. The VMM of

Bennett is able to transfer processing of interrupts to the preferred virtual machine. Paragraph [0054] of Bennett discusses the transfer of control to the non-preferred virtual machine, in which an interrupt controller interface logic is able to determine that the non-preferred virtual machine does manage interrupts, and control is transferred to the VMM as a result. These teachings of Bennett also do not provide any hint of a VMM to commence virtualization of an I/O device at that time.

In view of the foregoing, it is respectfully submitted that the § 102 rejection of claim 1 and its dependent claims is erroneous.

Reversal of the final rejection of the above claims is respectfully requested.

2. Claim 14.

Claim 14 depends from claim 1 and is therefore allowable for at least the same reasons as claim 1. Moreover, claim 14 recites that the virtual machine monitor temporarily pauses an I/O sequence by emulating the I/O device as being busy. With respect to the subject matter of claim 14, the Examiner cited ¶ [0054], lines 10-14 as purportedly disclosing the feature of claim 14. The cited passage in ¶ [0054] of Bennett refers to an interrupt controller interface logic that sets the interrupt flag to a value indicating that all interrupts are masked to prevent delivery of interrupts to the VMM. This action by the interrupt controller interface logic is different from the VMM temporarily pausing an I/O sequence by emulating the I/O device as being busy. Therefore, claim 14 is further allowable for the foregoing reason.

Reversal of the final rejection of the above claim is respectfully requested.

3. Claim 16

Claim 16 depends from claim 1 and is therefore allowable for at least the same reasons as claim 1. Moreover, claim 16 recites "devirtualizing the I/O device at runtime following the runtime virtualization." With respect to this subject matter of claim 16, the Examiner cited ¶ [0055], lines 1-5, of Bennett. This passage of Bennett refers to transitioning control to the VMM, where the VMM is notified that the cause of the transfer is the pending interrupt. The cited passage Bennett also states that the VMM, knowing that all the interrupts are to be handled by the preferred virtual machine, modifies the interrupt control indicators to allow the preferred virtual machine to manage all interrupts and transfers control to the preferred virtual machine. This teaching of Bennett has nothing to do with virtualizing an I/O device at runtime following runtime virtualization.

Therefore, claim 16 is further allowable for the foregoing reason. Reversal of the final rejection of the above claim is respectfully requested.

4. Claims 47-51, 54, 55.

Independent claim 47 is also not anticipated by Bennett, since Bennett fails to disclose computer-readable memory encoded with software for causing the computer to **commence** virtualization of the I/O device at **runtime**. As explained above in connection with claim 1, the concept of commencing virtualization of an I/O device at runtime clearly does not exist in Bennett. Therefore, claim 47 and its dependent claims are allowable over Bennett.

Reversal of the final rejection of the above claims is respectfully requested.

5. Claim 53.

Claim 53 depends from claim 52. Note that claim 52 was rejected as obvious over Bennett, whereas claim 53 (which depends from claim 52) was rejected as being anticipated by Bennett. Such a rejection clearly is defective on its face, since the Examiner necessarily must concede that Bennett fails to disclose the subject matter of claim 53, which incorporates the subject matter of claim 52.

Moreover, as explained above in connection with claim 14, Bennett clearly fails to disclose a virtual machine monitor temporarily pausing an I/O sequence by emulating the I/O device as being busy.

Reversal of the final rejection of the above claim is respectfully requested.

6. Claims 17, 18, 25, 28, 56, 65, 66.

Independent claim 17 was also rejected as purportedly anticipated by Bennett. Claim 17 recites a virtual machine monitor running on the hardware of a computer, an operating system running on the virtual machine monitor, and the hardware including an I/O device, the I/O device already virtualized by the virtual machine monitor, and the I/O device being devirtualized at runtime.

As explained above in connection with claim 16, ¶ [0055], lines 1-5, of Bennett does not provide any teaching of devirtualizing an I/O device at runtime. The Examiner argued that the VMM of Bennett modifying the interrupt control indicator to allow the preferred virtual machine to manage interrupts constitutes devirtualization—such a reading is clearly erroneous. A virtual machine managing interrupts actually **continues** the virtualization, and cannot possibly be construed as devirtualizing an I/O device.

Therefore, claim 17 and its dependent claims are clearly allowable over Bennett.

Independent claim 56 and its dependent claims are also similarly allowable over Bennett.

Reversal of the final rejection of the above claims is respectfully requested.

7. Claims 19, 57, 61.

Claims 19, 57, and 61 depend from respective base claims 17 and 56, and therefore are allowable for at least the same reasons as corresponding base claims.

Moreover, claim 19 recites that the devirtualization includes stopping I/O device emulation at runtime. With respect to claim 19, the Examiner referred to ¶ [0054], lines 1-4, of Bennett, which refers to transferring control to the non-preferred virtual machine, with the VMM setting an interrupt control indicator to a value indicating that the non-preferred virtual machine does not manage any interrupts. There is nothing in this passage of Bennett to even remotely hint at devirtualizing the I/O device at runtime that includes stopping I/O device emulation at runtime.

Claim 57 and its dependent claim 61 are further allowable for similar reasons.

Reversal of the final rejection of the above claim is respectfully requested.

8. Claims 26, 63.

Claims 26 and 63 depend from respective base claims 17 and 56, and are therefore allowable for at least the same reasons as respective base claims. Moreover, claim 26 recites configuring the hardware so the accesses by the operating system to the I/O device no longer trap to the virtual machine monitor. With respect to claim 26, the Examiner cited ¶ [0055] of Bennett as purportedly disclosing the subject matter of the claim. Paragraph [0055] of Bennett refers to transitioning control to the VMM, and notifying the VMM that the cause of the transfer is a pending interrupt. The cited paragraph of Bennett also notes that the VMM knows that all interrupts are to be handled by the preferred virtual machine, and the VMM modifies the interrupt control indicators to allow the preferred virtual machine to manage all interrupts and transfers control to the preferred virtual machine. Nowhere in this passage of Bennett is there any hint of configuring hardware so operating system accesses to the I/O device do not trap to the virtual machine monitor.

Therefore, claim 26 is further allowable for the foregoing reasons. Claim 63 is further allowable for similar reasons. Reversal of the final rejection of the above claims is respectfully requested.

9. Claims 29-34, 37.

Independent claim 29 recites a computer that comprises hardware including an I/O device, and computer memory encoded with a virtual machine for running on the hardware and commencing virtualization of the I/O device at runtime. As explained in connection with claim 1, Bennett does not provide any teaching of a virtual machine for running on the hardware and **commencing** virtualization of the I/O device at **runtime**.

Therefore, claim 29 and its dependent claims are clearly allowable over Bennett.

Reversal of the final rejection of the above claims is respectfully requested.

10. Claim 36.

Claim 36 depends indirectly from claim 29 and is therefore allowable for at least the same reasons as claim 29. Moreover, claim 36 recites the virtual machine monitor temporarily pausing the I/O sequence by emulating the I/O device as being busy. As explained above in connection with claim 14, Bennett clearly does not disclose the above subject matter.

Claim 36 is therefore further allowable for the foregoing reasons. Reversal of the final rejection of the above claim is respectfully requested.

11. Claims 38, 46.

Independent claim 38 recites a computer comprising hardware including an I/O device, and computer memory encoded with a virtual machine monitor for devirtualizing the I/O device at runtime. As explained above in connection with claim 17, there is no concept in Bennett of a virtual machine monitor for devirtualizing the I/O device at runtime.

Therefore, claims 38 and its dependent claims are clearly allowable over Bennett. Reversal of the final rejection of the above claims is respectfully requested.

12. Claim 44.

Claim 44 depends from claim 38 and is therefore allowable for at least the same reasons as claim 38. Moreover, as explained above in connection with claims 26 and 63, Bennett does not provide any teaching of configuring hardware so operating system accesses to the I/O device no longer trap to the virtual machine monitor.

Therefore, reversal of the final rejection of the above claim is respectfully requested.

B. Claims 9, 11, 15, 20-24, 27, 35, 39-43, 45, 52, 58-62, and 64 Rejected Under 35 U.S.C. § 103(a) as Unpatentable Over Bennett Alone.

1. Claim 9.

In view of the defective rejection of the base claim 1 over Bennett, it is respectfully submitted that the obviousness rejection of claim 9 over Bennett is also defective.

Claim 9 further recites that the virtual machine monitor commences emulation by intercepting I/O accesses where the virtual machine monitor uses the intercepted I/O accesses to update a state machine, whereby the state machine reflects a state of the I/O device, and where the virtual machine monitor examines transitions in the state machine to determine whether the I/O device is in the middle of an I/O sequence.

The Examiner conceded that this subject matter of claim 9 is not found in Bennett. 8/8/2008 Office Action at 8. However, the Examiner concluded that this feature would have been obvious, since a person of ordinary skill "would be motivated by the desire to have a method of representing the state of the I/O device to track its operation." *Id.* The conclusion reached by the Examiner is not based on any objective evidence, but rather appears to be based on the teachings of the present invention itself. Bennett provides absolutely no hint whatsoever of the subject matter of claim 9. Therefore, the obviousness rejection of claim 9 is clearly defective.

Reversal of the final rejection of the above claim is respectfully requested.

2. Claims 11, 35, 52.

In view of the allowability of base claims 1, 29, and 47 over Bennett, it is respectfully submitted that the obviousness rejection of claims 11, 35, and 52 over Bennett has been overcome.

Claim 11 recites that the virtual machine monitor uses a state machine to determine whether the I/O device is in the middle of an I/O sequence, and delays commencing emulation until the state machine indicates that I/O sequence has completed.

The Examiner conceded that Bennett fails to disclose the subject matter of claim 11, but concluded, without citing to any objective evidence, that the subject matter of claim 11 would be obvious. Bennett provides absolutely no hint of delaying commencing emulation until the state machine indicates that an I/O sequence has completed. Therefore, claim 11 is clearly non-obvious over Bennett.

Claims 35 and 52 are similarly allowable over Bennett.

Reversal of the final rejection of the above claim is respectfully requested.

3. Claims 15, 20, 22, 23, 24, 27, 39, 41-43, 45, 58, 60-62, 64.

In view of the allowability of base claims over Bennett, the obviousness rejection of the above claims has been overcome. Reversal of the final rejection of the above claims is respectfully requested.

4. Claims 21, 40, 59.

In view of the allowability of base claims over Bennett, it is respectfully submitted that the obviousness rejection of dependent claims 21, 40, and 59 over Bennett has also been overcome.

Claim 21 recites that the virtual machine monitor temporarily stops the operating system by emulating the I/O device as being in a "busy" or "device not ready" state. The Examiner cited ¶ [0054], lines 10-14, as purportedly disclosing this feature of claim 21. The cited passage of Bennett refers to the interrupt controller interface logic setting the interrupt flag to a value indicating that all interrupts are masked to prevent delivery of interrupts to the VMM. There is no hint given in this passage of the virtual machine monitor temporarily stopping the operating system by emulating the I/O device as in a "busy" or "device not ready" state.

Dependent claims 40 and 59 are similarly allowable over Bennett.

Reversal of the final rejection of the above claims is respectfully requested.

CONCLUSION

In view of the foregoing, reversal of all final rejections and allowance of all pending claims is respectfully requested.

Respectfully submitted,

Date: Jan 9, 2009

[illegible]

Dan C. Hu
Registration No. 40,025
TROP, PRUNER & HU, P.C.
1616 South Voss Road, Suite 750
Houston, TX 77057-2631
Telephone: (713) 468-8880
Facsimile: (713) 468-8883

VIII. APPENDIX OF APPEALED CLAIMS

The claims on appeal are:

1. In a computer including an I/O device, a method comprising using a virtual machine monitor to commence virtualization of the I/O device at runtime.

2. The method of claim 1, wherein the computer further includes a CPU, wherein the virtual machine monitor is in control of the CPU prior to the runtime virtualization of the I/O device.

3. The method of claim 1, wherein the virtualization is performed transparently to the operating system.

4. The method of claim 1, wherein the I/O device is compatible with the virtualized I/O device.

5. The method of claim 1, wherein the virtualization includes commencing I/O device emulation at runtime.

6. The method of claim 5, further comprising configuring the hardware to trap I/O accesses, and enabling the virtual machine monitor to emulate the I/O device in response to the traps.

7. The method of claim 6, wherein the virtual machine monitor uses memory management to trap the I/O accesses.

8. The method of claim 5, wherein the virtual machine monitor can commence the emulation between I/O sequences.

9. The method of claim 8, wherein the virtual machine monitor commences emulation by intercepting I/O accesses; wherein the virtual machine monitor uses the intercepted I/O accesses to update a state machine, whereby the state machine reflects a state of the I/O device; and wherein the virtual machine monitor examines transitions in the state machine to determine whether the I/O device is in the middle of an I/O sequence.

10. The method of claim 5, wherein the virtual machine monitor can commence the emulation in the middle of an I/O sequence.

11. The method of claim 5, wherein the virtual machine monitor uses a state machine to determine whether the I/O device is in the middle of an I/O sequence, and delays commencing emulation until the state machine indicates that I/O sequence has completed.

12. The method of claim 1, wherein the runtime virtualization includes using the virtual machine monitor to emulate I/O device interrupts.

13. The method of claim 1, wherein I/O device interrupts are directed to an operating system prior to the runtime virtualization of the I/O device; and wherein the I/O device interrupts are directed to the virtual machine monitor during and after the virtualization of the I/O device.

14. The method of claim 1, wherein the virtual machine monitor temporarily pauses an I/O sequence by emulating the I/O device as being busy.

15. The method of claim 1, wherein the I/O device has multiple modes of operations; wherein the virtual machine monitor determines the mode of the I/O device prior to commencing virtualization; and wherein the virtual machine monitor restores the determined mode of the operation after virtualization.

16. The method of claim 1, further comprising devirtualizing the I/O device at runtime following the runtime virtualization.

17. In a computer including hardware, a virtual machine monitor running on the hardware, an operating system running on the virtual machine monitor, the hardware including an I/O device, the I/O device already virtualized by the virtual machine monitor, a method comprising devirtualizing the I/O device at runtime.

18. The method of claim 17, wherein the devirtualization is performed transparently to the operating system.

19. The method of claim 17, wherein the devirtualization includes stopping I/O device emulation at runtime.

20. The method of claim 17, wherein the virtual machine monitor emulates the I/O device prior to devirtualization; and wherein the devirtualization includes allowing the virtual machine monitor to temporarily stop the operating system from commencing a new I/O sequence.

21. The method of claim 20, wherein the virtual machine monitor temporarily stops the operating system by emulating the I/O device as being in a "busy" or "device not ready" state.

22. The method of claim 20, wherein the virtual machine monitor bounds the amount of time the operating system processing is temporarily stopped.

23. The method of claim 20, wherein the virtual machine monitor logs I/O accesses by the operating system to the I/O device during devirtualization, and replays the log to the device after devirtualization, whereby the I/O accesses by the operating system are deferred during the devirtualization of the I/O device.

24. The method of claim 17, wherein the virtual machine monitor waits for I/Os initiated by the virtual machine monitor's driver for the I/O device to complete, and for all expected interrupts from the device to arrive, before ceasing device emulation.

25. The method of claim 17, further comprising re-directing interrupts from interrupt handlers in the-virtual machine monitor to interrupt handlers in the operating system.

26. The method of claim 17, further comprising configuring the hardware so the accesses by the operating system to the I/O device no longer trap to the virtual machine monitor.

27. The method of claim 17, wherein the I/O device has multiple modes of operations; wherein the virtual machine monitor determines the mode of the I/O device prior to commencing devirtualization; and wherein the virtual machine monitor restores the determined mode of the operation after devirtualization.

28. The method of claim 17, wherein the I/O device is virtualized at runtime again after having been devirtualized at runtime.

29. A computer comprising:

hardware including an I/O device; and

computer memory encoded with a virtual machine for running on the hardware and commencing virtualization of the I/O device at runtime.

30. The computer of claim 29, wherein the I/O device is compatible with the virtualized I/O device.

31. The computer of claim 29, wherein the virtualization includes commencing I/O device emulation at runtime.

32. The computer of claim 31, further comprising configuring the hardware to trap I/O accesses, and enabling the virtual machine monitor to emulate the I/O device in response to the traps.

33. The computer of claim 32, wherein the virtual machine monitor uses memory management to trap the I/O accesses.

34. The computer of claim 31, wherein the virtual machine monitor can commence the emulation in the middle of an I/O sequence.

35. The computer of claim 34, wherein the virtual machine monitor uses a state machine to determine whether the I/O device is in the middle of an I/O sequence, and delays commencing emulation until the state machine indicates that I/O sequence has completed.

36. The computer of claim 31, wherein the virtual machine monitor temporarily pauses the I/O sequence by emulating the I/O device as being busy.

37. The computer of claim 29, wherein the runtime virtualization includes using the virtual machine monitor to emulate I/O device interrupts.

38. A computer comprising:

hardware including an I/O device; and

computer memory encoded with a virtual machine monitor for devirtualizing the I/O device at runtime.

39. The computer of claim 38, wherein the virtual machine monitor emulates the I/O device prior to commencing devirtualization; and wherein the virtual machine commences the devirtualization by temporarily stopping an operating system running on the virtual machine monitor from commencing a new I/O sequence.

40. The computer of claim 39, wherein the virtual machine monitor temporarily stops the operating system by emulating the I/O device as being in a "busy" or "device not ready" state.

41. The computer of claim 39, wherein the virtual machine monitor bounds the amount of time the operating system processing is temporarily stopped.

42. The computer of claim 39, wherein the virtual machine monitor logs I/O accesses by an operating system to the I/O device during devirtualization, and replays the log to-the device after devirtualization.

43. The computer of claim 39, wherein the virtual machine monitor waits for I/Os initiated by a virtual machine monitor driver for the I/O device to complete, and for all expected interrupts from the I/O device to arrive, before ceasing device emulation.

44. The computer of claim 38, further comprising configuring the hardware so operating system accesses to the I/O device no longer trap to the virtual machine monitor.

45. The computer of claim 38, wherein the I/O device has multiple modes of operations; wherein the virtual machine monitor determines the mode of the I/O device prior to commencing devirtualization; and wherein the virtual machine monitor restores the determined mode of the operation after the I/O device has been devirtualized.

46. The computer of claim 38, wherein the virtual machine monitor can virtualize the I/O device after having devirtualized the I/O device at runtime.

47. An article for a computer including an I/O device, the article comprising computer-readable memory encoded with software for causing the computer to commence virtualization of the I/O device at runtime.

48. The article of claim 47, wherein the virtualization includes commencing I/O device emulation at runtime.

49. The article of claim 48, wherein the software includes a virtual machine monitor; and wherein the software configures the hardware to trap I/O accesses, and enables the virtual machine monitor to emulate the I/O device in response to the traps.

50. The article of claim 49, wherein the virtual machine monitor uses memory management to trap the I/O accesses.

51. The article of claim 48, wherein the software includes a virtual machine monitor for commencing the emulation in the middle of an I/O sequence.

52. The article of claim 51, wherein the virtual machine monitor includes a state machine for determining whether the I/O device is in the middle of an I/O sequence, the virtual machine monitor delaying the commencement of the emulation until the state machine indicates that the I/O sequence has completed.

53. The article of claim 52, wherein the virtual machine monitor temporarily pauses the I/O sequence by emulating the I/O device as being busy.

54. The article of claim 47, wherein the software includes a virtual machine monitor for emulating I/O device interrupts during the runtime virtualization.

55. The article of claim 47, wherein the software includes a virtual machine monitor for commencing the virtualization of the I/O device at runtime.

56. An article for a computer including an I/O device, the article comprising computer-readable memory encoded with software for causing the computer to devirtualize the I/O device at runtime.

57. The article of claim 56, wherein the devirtualization includes ceasing emulation of the I/O device at runtime.

58. The article of claim 57, wherein the software includes a virtual machine monitor; and wherein the devirtualization includes temporarily stopping an operating system running on the virtual machine monitor from commencing a new I/O sequence.

59. The article of claim 58, wherein the virtual machine monitor temporarily stops the operating system by emulating the I/O device as being in a "busy" or "device not ready" state.

60. The article of claim 58, wherein the virtual machine monitor bounds the amount of time the operating system processing is temporarily stopped.

61. The article of claim 57, wherein the software includes a virtual machine monitor for ceasing the emulation; the virtual machine monitor waiting for I/Os initiated by a virtual machine monitor driver for the I/O device to complete, and for all expected interrupts from the I/O device to arrive, before ceasing device emulation.

62. The article of claim 56, wherein the software includes a virtual machine monitor for logging I/O accesses by an operating system to the I/O device during devirtualization, and replaying the log to the I/O device after devirtualization.

63. The article of claim 56, wherein the software includes a virtual machine monitor, the software configuring the hardware so operating system accesses to the I/O device do not trap to the virtual machine monitor.

64. The article of claim 56, wherein the I/O device has multiple modes of operations; and wherein the software includes a virtual machine monitor for determining the mode of the I/O device prior to commencing devirtualization; and restoring the determined mode of the operation after the I/O device has been devirtualized.

65. The article of claim 56, wherein the software includes a virtual machine monitor for devirtualizing the I/O device at runtime.

66. The article of claim 65, wherein the virtual machine monitor can virtualize the I/O device after having devirtualized the I/O device at runtime.

IX. EVIDENCE APPENDIX

None.

X. <u>RELATED PROCEEDINGS APPENDIX</u>

None.